AD_____

Award Number: W81XWH-05-1-0410

TITLE: Pre-Clinical Validation of Robotic Scrub Technician

PRINCIPAL INVESTIGATOR: Michael R. Treat, M.D.

CONTRACTING ORGANIZATION: Robotic Surgical Technology, Inc.
New York, NY 10034-1159

REPORT DATE: Jan 2006

TYPE OF REPORT: Final

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
01-01-2006	Final	13 JUN 2005 - 12 DEC 2005

4. TITLE AND SUBTITLE
Pre-Clinical Validation of Robotic Scrub Technician

5a. CONTRACT NUMBER

5b. GRANT NUMBER
W81XWH-05-1-0410

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Michael R. Treat, M.D.

E-mail: mt23@columbia.edu

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Robotic Surgical Technology, Inc.
New York, NY 10034-1159

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
Original contains color plates: All DTIC reproductions will be in black and white.

14. ABSTRACT

The purpose of this work is to develop a robotic scrub technician for use in the surgical operating room. The robot consists of a computer connected to a voice recognition system, a digital camera and a mechanical arm with a gripper to handle surgical instruments. When the surgeon verbally requests an instrument, the mechanical arm delivers that instrument to the surgeon's hand. The robot uses machine-vision via the digital camera to locate and identify surgical instruments so that they can be properly retrieved from the surgical field. The scope of this work is to achieve specific technical goals relating to the robotic machine-vision system, the hardware of the system and the overall integration and validation testing of the system. We have achieved the software development goals relating to the machine-vision system, and the hardware design and construction goals. We have begun the overall system validation process but it is not complete at this time. This work is significant because it contributed to the attainment of a clinically useful robotic scrub technician system.

15. SUBJECT TERMS
machine-vision, surgical robotics, artificial intelligence, autonomous, surgical instrument

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
					USAMRMC
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*
U	U	U	UU	40	

Table of Contents

INTRODUCTION

The purpose of this work is to develop a robotic scrub technician for use in the surgical operating room. The robot consists of a computer connected to a voice recognition system, a digital camera and a mechanical arm with a gripper to handle surgical instruments. When the surgeon verbally requests an instrument, the mechanical arm delivers that instrument to the surgeon's hand. The robot uses machine-vision via the digital camera to locate and identify surgical instruments so that they can be properly retrieved from the surgical field. The scope of this work is to achieve specific technical goals relating to the robotic machine-vision system, the hardware of the system and the overall integration and validation testing of the system. The work described herein is the finalizing of the technical development of the machine before its initial clinical test case in the operating room.

The significance is this work is several-fold. Such a device would be logistically better for forward deployed military hospitals since it would reduce the number of personnel that are required to deliver surgical care. For civilian and military hospitals, the device would be clinically better by reducing errors in counting/tracking instruments and supplies used in surgery. The device would improve overall operating room throughput by freeing up personnel to perform the numerous tasks that have to be done to get an operating room ready for the next case. By using this robot, an operating room could perform more cases per day with the same number of personnel. Robotic surgical assistants will improve the quality, consistency and portability of surgical care for civilians, soldiers and space travelers.

BODY

From our **Statement of Work** from the original proposal:

"STATEMENT OF WORK
The proposed work is in two parts: The first part is achieving several technical objectives for the machine-vision system that will enable the robot to perform certain functions that we anticipate will be helpful in the performance of its clinical duties. The second part is the validation of the machine-vision system as part of the full robotic system in a realistic surgical operating environment on a inanimate model. Successful completion of the goals of the first part of the S.O.W. will set the stage for the second part of the work.

Part 1. Achieving Technical Objectives

These are the technical objectives that we consider useful for the robotic system's clinical performance.

Dual Camera Integration: Two digital cameras will be linked in software to expand the visual field of the machine-vision system in order to provide full visual coverage of the surgical field and all instrument-bearing trays and transfer surfaces.

Arm Masking Out: The machine-vision system will be given the ability to deal with the image of the robot's own arm moving in its visual field.

Auto Adjust Camera: The machine-vision system will be given the ability to self-adjust its camera parameters to obtain best feature recognition of items in its visual field.

Motion Detector: The machine-vision system will be given the ability to detect motion of an object in its visual field.

Multiple Simultaneous Object Identification: The machine-vision system will be given the ability to identify multiple objects simultaneously in its visual field.

Design/Construction of Instrument ("Mayo") tray and Transfer zone: Instrument-bearing trays and surfaces will be designed and fabricated in order to meet the functional, performance and user-interface requirements of the robotic system.

Design/Construction of System Stand: A system stand for the robotic arm, instrument-bearing trays and surfaces and camera system will be designed and fabricated to meet the functional, performance and user-interface requirements of the robotic system.

Part 2. Performing Pre-clinical Validation

Validation Process: The goal of the pre-clinical validation is to show that the machine-system that we are designing is appropriate for the proposed clinical use. The machine-vision system will be evaluated in the context of the entire robotic scrub technician system. The evaluation process will involve a full running of the entire system in a close approximation of clinical use. This validation has relevance for the machine-vision system as well as for the overall system."

The following Gantt Chart was taken from the original proposal:

GANTT CHART OF TASKS						
	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6
SOFTWARE TASKS						
Dual Camera Integration	XxxxxxxxX					
Arm Masking Out.		XxxxxxxxxxX				
Auto Adjusting Camera	XxxxxxxxxxxX					
Motion Detector.		xxxxxxxxxxxX				
Multiple Simultaneous Objects		XxxxxxxxxxX				
Software Integration				xxxxxxX		
HARDWARE TASKS						
Instrument Trays	XxxxxxxxxxxX					
System Stand			XxxxxxxxxxxxX			
Hardware integration				xxxxX		
VALIDATION TASK					xxxxxxxxxxxX	

At the present time, most of the technical tasks described above have been completed. Although the POP of this contract officially began in June, 2005, the work has actually been going on since before that time. These software and hardware technical tasks are synergistic with the goals of

our NSF Phase II SBIR grant, and to some extent, with our DARPA/TATRC contract for the Trauma Pod.

SOFTWARE TASKS

The software technical tasks that we regard as completed are:

COMPLETED SOFTWARE TASKS
Dual Camera Integration
Arm Masking Out.
Auto Adjusting Camera
Motion Detector
Multiple Simultaneous Objects
Software Integration

These tasks are mostly concerned with the machine-vision system of the robot.

The following sections show the results and discuss the methods of our completing of these tasks.

Dual Camera Integration

This task refers to creating the ability of the machine-vision system to "see" both of the instrument carrying surfaces on the robot, the instrument tray and the transfer zone. At the start of this work, the machine-vision (from a single camera) was being used only on the transfer zone so that the system can locate instruments placed there by the surgeon wherever the surgeon

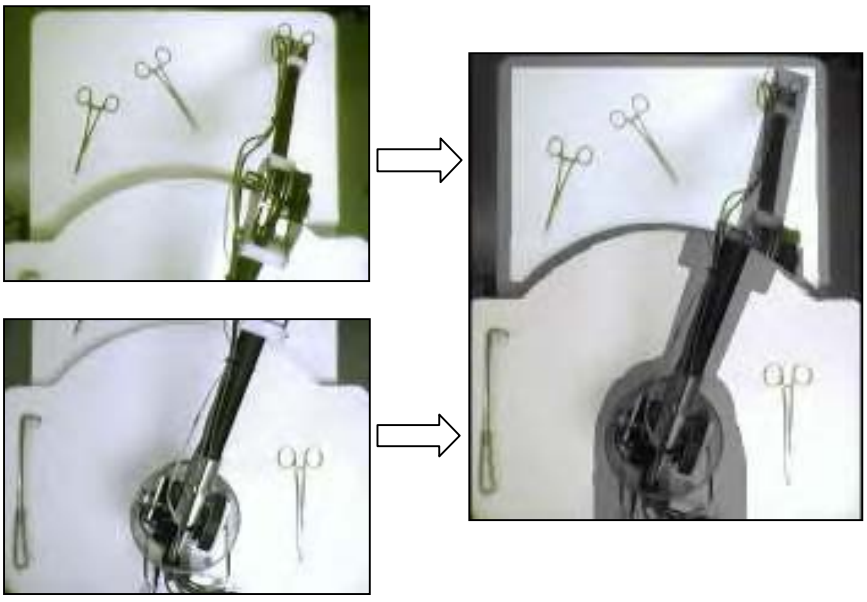

Fig. 1. Illustration of Dual Camera Integration and Arm Masking Out. On the left are two separate camera views from the two individual cameras. These views have been "stitched" together in the single picture on the right. There is an angular offset to the distal part of the arm that is corrected in the software. The mask has also been applied to the arm. This mask moves with the arm. The machine-vision software ignores the area covered by the mask. This prevents the machine-vision software from being confused into thinking that the robot's own arm is an instrument or other object.

happens to throw that instrument down. Being able to "see" the instrument tray will make the robot more resilient in terms of being able to recover from mechanically induced errors in the positioning of the instruments when they are returned to the instrument tray. We judged that it would be better to have two separate cameras rather than one camera with a wide-angle coverage, in order to maintain the same scale of view (meters per pixel) and hence resolution that we have with the single camera system. Hence, our machine-vision software was modified to incorporate visual data from two cameras and to be able to "stitch together" the inputs from each camera into one cohesive world-view.

The technical challenge here was to describe the mathematical coordinate transformations required to link the two pictures together into a single coordinate system. Referring to Fig 6, one can see that the two cameras are located at different positions on the camera post. This means that the coordinate systems for each of the two cameras are different, since the origin position of each camera is different.

The basic method that was used to reconcile the two coordinate systems was to maintain two separate mappers for camera coordinates to real-world coordinates. When an instrument is seen by the cameras, we first determine which camera it was seen by. This is not trivial because in the stitched image the distinction is lost, and the pattern by which the cameras were stitched together may be arbitrarily complicated, and not simply separable by a horizontal or vertical line. Therefore we look to the 'stitch pattern' itself to find which zone the object falls in. Once we have this information we can use the mapper for that camera, which takes into account the camera's X-Y location (the point in the trays over which it is centered) and the Z distance to the tray, which determines the meter-per-pixel scale. The X-Y location of the instrument in real-world coordinates is given by the mapper's equation, while the Z is given by the known position of the tray the instrument is sitting on, plus the known height of the instrument once it has been identified. This completes the three-dimensional real-world position of the instrument, which can then be used in a command to send the arm to pick it up.

Arm Masking Out

This task refers to creating the ability of the machine-vision software to effectively screen out and ignore the robot's arm when the arm is moving over the transfer zone/Mayo tray. At the start of this work, the machine-vision software that is scanning the transfer zone for an instrument was turned off whenever the arm moves over the transfer zone. This imposes a fairly severe performance penalty in terms of the robot's ability to retrieve instruments from the transfer zone on a timely basis. By developing an "arm masking out" capability, the object scanning process can continue even when the arm is moving over the transfer zone. Masking is also necessary to take advantage of the view of the Mayo stand that Dual Camera Integration will provide, because the arm is nearly always partially obstructing the Mayo stand. This task involves integrating information from the motion control system about where the arm is so that the machine-vision software can "ignore" objects with a bounding box centered at the arm's known location in space.

The Arm Mask was implemented as a set of polygons that are created as part of the software's physical model of the robot. They appear as a set of rectangles that are each attached to a segment of the arm, and the Simulator software that updates the position of the arm as part of the robot's control loop then also updates the mask polygons. In this way it is assured that the mask

will always be up to date with the position of the arm. The task that remains is then to convert the real-world position of the mask polygons into camera coordinates and overlay them onto the camera image. This is the reverse of the conversion that we perform to determine where instruments that have been seen by the camera are in real-world coordinates, but it is complicated by the fact that different points on the arm are at different heights, unlike the instruments which all sit on the flat trays, with negligible differences in height. Because the robot is moving in three dimensions, we must account for the perspective of the camera. This was accomplished with a perspective equation that adjusts the X and Y coordinates of the points in the polygons according to their Z.

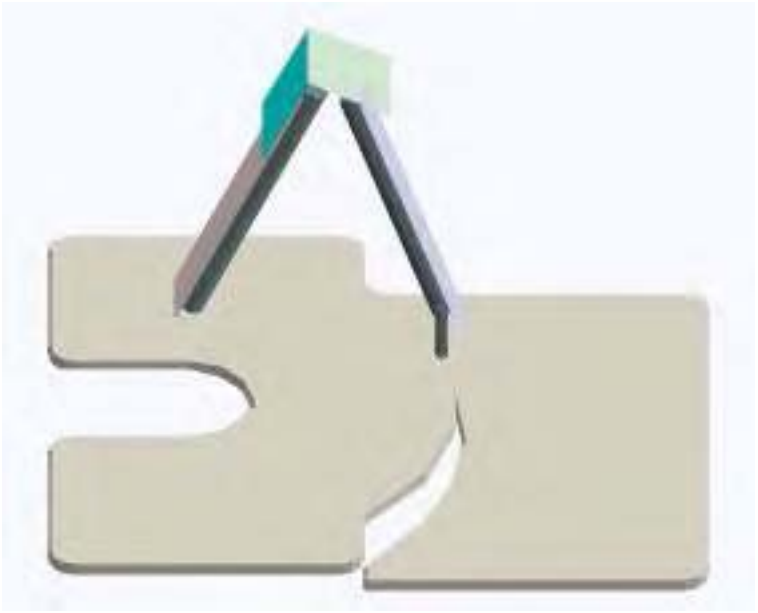

Fig. 2. The arm mask polygons as they appear in the software's simultor viewer, hovering slightly apart from the representation of the arm itself.

There was another technical challenge that came up as part of this task, involving the system's maintenance of knowledge about the state of the instruments seen by the cameras. Up to this point, any time an image was taken and analyzed, it was regarded as a reliable source of knowledge about the state of all the instruments being worked with. Therefore, if an instrument was missing from the image, the system no longer knew where it was. But with the arm mask it is possible for an instrument to be hidden underneath the arm. In this case we must be able to ignore the fact that the instrument was not seen, and wait until it is next revealed to update its position.

This problem was solved in two parts. First we implemented an "invisible arm" mechanism, by which any pixels within the arm mask are taken from the previous image. This means that any time we look at an area hidden by the arm, we use visual information from the last time that area was visible to fill in the image. The result is that if we take pictures while sweeping the arm over a tray full of instruments, it will appear as if the arm is invisible, because the areas that are hidden in the current image will continuously be filled in by information from previous images (as long as the arm has moved around enough that all pixels have been revealed at least once). This solution works alone as long as the old information being used to fill in the images is still valid, but once the instruments are being moved around, something else is needed. For example,

if the arm picks up an instrument, delivers it, and then returns to the area over where that instrument had been, and a picture is taken, the instrument will still be seen where it was, due to the old information being used to fill in the image. The result would be that the system would think the arm had failed to pick up the instrument.

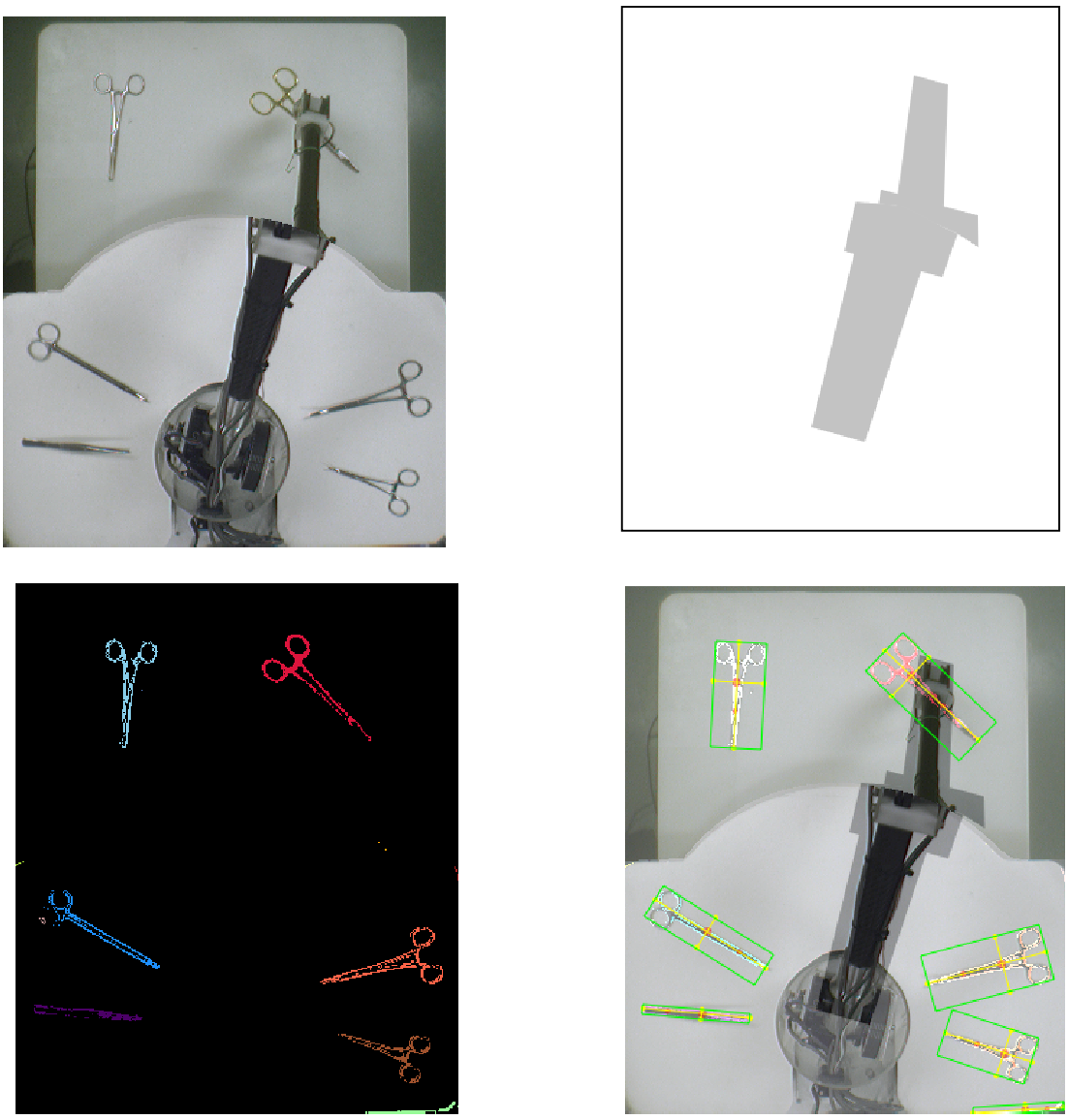

Fig. 3 These images show the "invisible arm" process by which objects that are partially hidden by the arm can still be measured and identified by the vision system. At top left is a raw image in which the arm is hiding part of one instrument, at top right the arm mask polygons are projected onto the image, at bottom left are the blobs, and at bottom right are the mask,raw image and blobs overlaid, showing that the hidden instrument was properly observed and measured. The pixels inside the arm mask were taken from a previous image in which they were not hidden.

We also needed a way to determine whether an object seen by the cameras was within the part of the image based on old information, i.e. within the arm mask. It was necessary that the implementation avoid computationally expensive algorithms involving construction of polygons to fit the collection of pixels that is our view of an object, or polygon intersection tests. We settled on a simple but effective test involving a small set of points that we already acquire in measuring the object, and checking whether any of them were inside the arm mask. If so, we flag the object as being based on possibly outdated information. The rules in the cognitive architecture that govern tracking instruments as they move through the system then have the option of refusing to base a judgment on an object that was obscured by the arm. The key is that under nominal conditions, it may be reasonable to rely on old information, but in more unusual circumstances (such as deciding that the arm has failed to pick up an instrument) it is necessary to demand information that is less in doubt. The result is that we are able to take advantage of the 'invisible arm' without putting too much faith in it.

Auto Adjusting Camera

As detailed in our proposal, this task refers to enabling the machine-vision software to programmatically adjust the camera parameters so as to obtain the best feature recognition of the instruments. This is the adjustment of camera parameters such as brightness, white balance, and saturation to obtain the most intact and well-defined segmentation (separation) of the instruments from the background surface and visual noise. Examples of visual noise include shadows and blood-staining.

The difficulty of this task came from several directions. First, we had to decide which of the camera's parameters to use in the optimization. We also needed to decide what characteristics in an image would define an optimal set of camera settings. Finally, we had throughout the process to consider the efficiency of the optimization algorithm, because setting a single parameter on a camera takes a nontrivial amount of time, making it impractical to search through the entire space of settings.

In choosing the specific parameters to use in the optimization algorithm, and at the same time defining the qualities of an optimal image, we determined that the relevant characteristics of an image for our purposes are luminosity and color balance. Within luminosity, referring to the average brightness of the pixels in the image, we could choose to adjust the Brightness, Auto Exposure, or Gain parameters. Each affects luminosity differently. For color balance, meaning the average hue of the pixels, we adjust White Balance, which itself consists of 'UR' and 'VB' parameters.

The algorithm that was arrived at divides the optimization process into three stages. In the first stage, we optimize the Auto Exposure parameter to achieve a reasonable luminosity in the image. This is an attempt to ensure that before the algorithm gets any further we are not attempting to analyze a completely dark or far too bright image for anything more subtle, such as color balance. We also use a loose tolerance here to reduce the amount of time taken up by this stage. Auto Exposure was selected for this stage because it appeared to be a relatively powerful parameter for changing the luminosity of the image, and did so without washing out the image

too much. In the next stage, the White Balance parameter is used to optimize the color balance of the image, aiming for a certain 'Redness', 'Greenness', and 'Blueness'. Finally, the luminosity of the image is adjusted again, this time using the Gain parameter, and aiming with greater precision at the desired value.

Within this top-level algorithm, each stage uses the same algorithm for optimization of a single parameter. This inner algorithm takes a parameter, a criterion by which to score the quality of resulting images, a score to aim for, and a tolerance. For example, in the first stage, we might want to optimize Auto Exposure with the criterion being the luminosity (brightness) of the image, a target score of 650 (luminosity has a range of 0 to 765), and a tolerance of 15. In the last stage we might instead choose to optimize Gain, and use a tolerance of 5. But the modularity of the inner algorithm ensures that we could at any time choose to change our target scores, tolerances, scoring criteria, or the order in which we optimize each parameter. For example, we could manually tweak the target scores if it turns out that different lighting conditions produce the effect that a darker or redder image produces the best results. The inner optimization routine functions like a hill-climbing algorithm with a binary search, attempting to determine the rate at which changing the parameter's value will affect the score according to the scoring criterion, and zero in on the desired value. As it gets closer, it may need to reevaluate the ratio of change in parameter value to change in score, since the originally calculated rate over the whole range of the parameter may not hold at finer scales. The algorithm also must give up if it reaches the maximum or minimum allowed value of the parameter without reaching the target score.

Motion Detector

This task refers to creating the ability of the machine-vision software to detect motion of an object over the transfer zone/Mayo tray. This capability will be a basic tool for several desired behaviors. Currently if the software takes a motion-blurred picture of a hand traveling over the tray, it will measure it as if it were an instrument, and the arm may try to pick it up. The crucial difference between hands and instruments from the software's point of view is that instruments that have been laid down do not move. Motion detection will therefore allow the software to either ignore objects that move around, or eventually to specifically look at them for hand-tracking purposes. Other important behaviors include collision avoidance with the surgeon's hand, and efficient image analysis. The latter refers to the fact that if no motion has occurred between one frame and the next, there is no need to go through the image processing routines again. Inversely, if the arm starts moving to pick up an instrument and then motion is detected, it is necessary to analyze a new image and determine whether the situation has changed before the arm continues trying to pick up an instrument that may no longer be there. Motion detection will effectively become the system's way of knowing when something has changed in the field of view and the situation needs to be reevaluated.

We implemented a motion detection capability using the vision system. This allows us to determine if any external movement is in progress around the instrument trays. In general, since our vision system's field of view is static and the surgical instruments don't move on their own, the detection of motion means someone has approached the robot and may be manipulating the surgical instruments. This indicates that we should wait for the motion to cease and then reevaluate the trays to see if any instruments have been moved, added, or removed.

We implemented the motion detector at a very low level in the vision system architecture. Our Firewire cameras are configured to capture frames of video data at a rate of 15 per second. In the C++ code that interfaces directly to the Firewire camera we added a motionFactor, calculated for each frame. This motionFactor is a measure of the general quality of the frame's video data. We then simply compare successive motionFactors for successive frames. If the difference between these two values is large enough, motion is detected.

Through experimentation we determined that a measure of the average redness of the image works well as a motionFactor,

$$\hat{R}_n = \frac{\sum_{i=1}^{n} R_i}{n}$$

This is the average redness, R, of all n pixels in the image. One implementation issue we had to address is that the numerator in this equation can grow too large to store in a 4-byte integer variable. We therefore use a cumulative (or inductive) average calculation,

$$\hat{R}_{i+1} = \frac{i \bullet \hat{R}_i \bullet R_{i+1}}{i + 1}$$

In this equation we calculate the average cumulative redness as we loop through each pixel based on that pixel's redness and the previous value for average cumulative redness. At the end of the loop we have the total average as in the first equation, but we avoid having to sum the red values (between 0 and 255) over all pixels.

Another motion detection issue focused on sensor attenuation tuning. We want the software to react but not overreact to motion. Consider the process of someone reaching into the vision system's field of view to return an instrument. We could potentially generate many motion detection events, 15 events per seconds, during the reach in. This is actually too much information to process, and it is mostly irrelevant. The essential information is the transition from the *no-motion* state to the *motion* state at the beginning of the reach in, and the transition from *motion* to *no-motion* at the end. We implemented a filter on the motion detector to restrict events to state changes so as to avoid overwhelming the rest of the system with unneeded information.

Multiple Simultaneous Objects

This task refers to creating the ability of the machine vision software to deal with more than one object at a time on the transfer zone. Up to this point, the software has always identified the biggest object on the transfer zone, then that instrument is removed, and the new biggest object is identified until all the instruments have been returned to the Mayo stand. This behavior, the reflexive return of instruments, will no longer be acceptable. Instead the system must have an overall awareness of the instruments without necessarily taking any action based on them. It also

must of course see all the instruments on the Mayo stand at all times. This requires a modification to the software to measure all objects and maintain the state of the set of instruments that has been recognized.

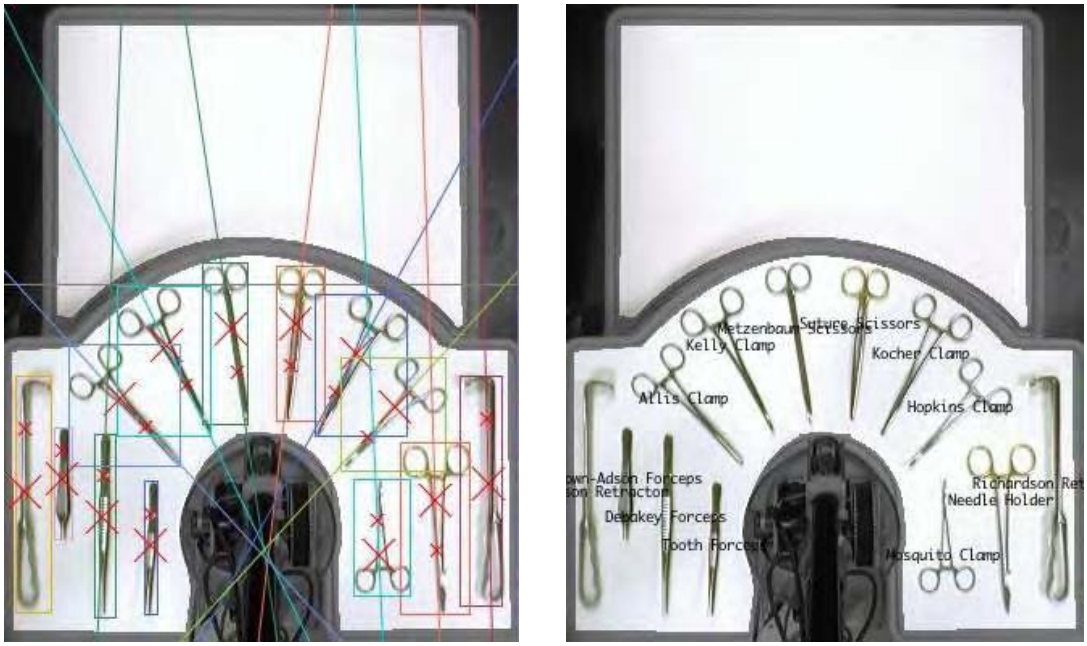

Fig.4. Demonstration of recognition of multiple simultaneous objects. The picture on the left shows that the machine vision software has correctly placed a "bounding box" on each instrument. The software has also found the centroid and "half-centroid" of each instrument, denoted by the large and small "x" marks. The software has also drawn a line through the axis of symmetry of each instrument. The picture on the right shows the identification labels correctly supplied by the software to each instrument.

This feature was implemented relatively simply by measuring and identifying all objects seen in an image rather than only the biggest one. The more difficult task was changing the behavior of the system in response to this new information. Where the robot previously returned any instrument seen on the transfer zone right away, and then simply remembered where instruments were supposed to be on the instrument tray, it now tracks the movements of all instruments throughout the system, and never picks up an instrument unless the vision system has reported seeing it in a precise position on one of the trays. This has hugely increased the situational awareness of the system, because the system has an idea of where each instrument is at all times, and can thus, among other abilities, provide intelligent responses when an instrument request cannot be fulfilled.

Another change introduced along with multiple object detection was an overhaul of the way blobs are measured and identified. Previously, a hard-coded algorithm performed the measurements and determined how they should be used in identifying an object. In the new

structure, new measurements can be added or taken away by writing new classes, and identifier classes can define new methods for aggregating the measurements and deciding on identifications. In this way, new measurements and identification algorithms can be quickly developed and tested for any improvement in distinguishing instruments.

Software Integration

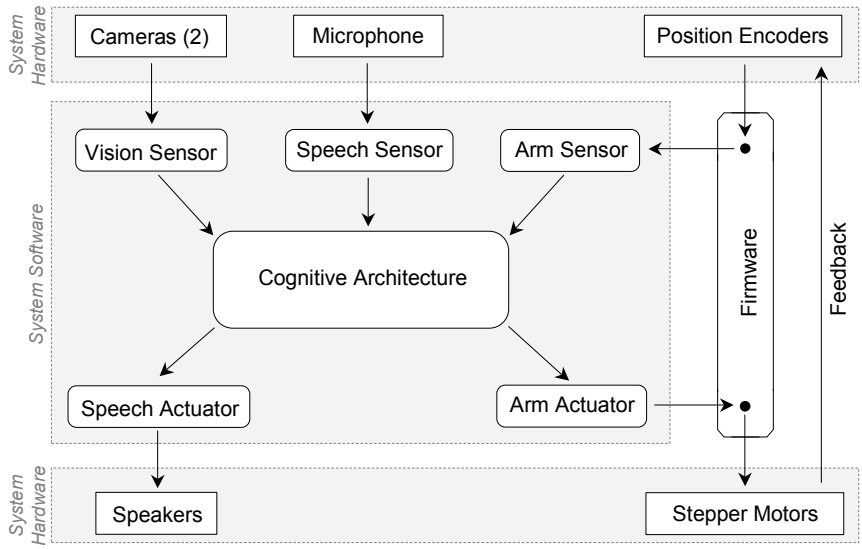

Fig. 5. Overall system software architecture is shown here. At the center is the cognitive architecture, which controls all behavior of the robot.

Software integration refers to bringing together all of the subsystems into a working whole. The final working integrated architecture of the system is shown in Fig. 5.

HARDWARE TASKS

The hardware technical tasks that we have completed are:

COMPETED HARDWARE TASKS
Instrument Trays
System Stand
Hardware integration

Fig 6. Final instrument trays were made of aluminum and covered with autoclavable mats with correct optical and mechanical properties.

Fig 7. Hardware integration is being done in this photo. The electronics enclosures are open for testing of the circuit boards within.

The final realization of the hardware integration is shown in Figure 8.

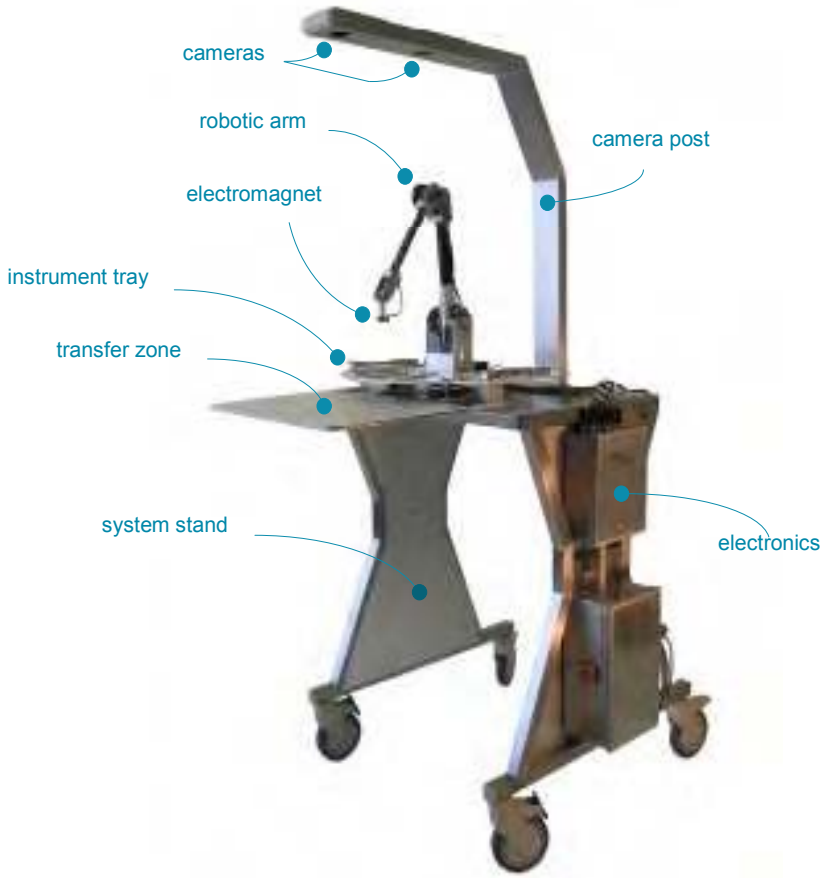

Fig 8. Final system hardware integration is shown here. This is a clinical grade robot.

VALIDATION TASK

The final part of the project is validation of the entire system. This is primarily a validation of the software, specifically the workings of the cognitive architecture. This is really a more difficult task than the others, since it is harder to define completion or success in an unambiguous way. What we are aiming for is that the robot be able to respond to the surgeon's needs in a clinically appropriate and helpful way. The cognitive architecture (CA) software is the part of the robot's overall software that we are concerned with here. This piece of software is fairly unique compared to the other parts of the robot's software, such as the machine-vision system or motor system software. What makes it different is that it is much harder to analyze this software by inspection of the code or by 'walking through" the various steps in the code. This difficulty arises because the CA software is designed to react to combinations of external inputs and internal states that the robot experiences. The CA software uses IF-THEN rules to determine the next step to take. When all the conditions of the IF clause are met, the rule "fires", activating the THEN clause of the rule. More than one rule may be activated at a given time if the right conditions exist to separately satisfy the IF clauses of all these rules. Also, the THEN clause of a

rule may satisfy the IF clause of yet another rule and thereby enable that rule to "fire". Even with a relatively modest number of rules (currently the system has 64 rules), there is a lot of potential for interaction and complexity. Thus, the software may produce outputs that were not anticipated. The only way to really debug this kind of system is to put the entire system through its paces, with as much life-like, random variation of conditions and inputs as is possible to achieve in the laboratory.

The rules are divided into Rule Sets as shown in the table below. This organization is important from a validation perspective in that it allows us to divide the process into largely independent modules that can be tested both individually and in combination with the other modules. In general, the flow moves from sensor expression to goal expression to goal actualization to step execution. There are several rules to accomplish each of these phases. The action of the rules in each phase acts as a trigger for the next phase rules. So sensor expression actions will trigger goal expression rules, and so on. We will now discuss each phase briefly.

Rule Set	Rules
autoParametersGoals	:optimizeStitchedCameraParameters :optimizeStitchedCameraParameters.preemptionHandler :optimizeNativeCameraParameters :testLuminosity
caRules	:runExpirationTimer :expireMarkedAssertion :ignoreDuplicateGoals
prioritizationRules	:prioritizeGoals :prioritize.noPreemption :preemptGoals :cancel
hardwareGoals	:initializeArm :correctPositionalDiscrepancy :calibrateStepper
instrumentCacheState	:recognizeBlob :findInstruments.blobsPerID :findInstruments.thatHaventMoved :findInstruments.withOnlyCacheKnown :findInstruments.inUseToTransferZone :findInstruments.unknownToITorTZ :findInstruments.initializeToIT :findInstruments.stillUnseen :findInstruments.transferZoneToInUse :findInstruments.intraCacheMove :findInstruments.movedBetweenITandTZ :findInstruments.inUseToInstrumentTray :findInstruments.missingBlob :findInstruments.noteComplete
instrumentGoals	:instrument.delivery

	:instrument.delivery.preemptionHandler :instrument.return :instrument.return.preemptionHandler :instrument.discard
sensorExpressionRules	:recognizedPhraseExpression :ignoreRecognizedButInappropriatePhrase :bumpSwitchExpression :motionDetectorExpression :periodicBlobDetection :armPositionExpression :correctPositionalDiscrepancyGoalExpression :controlSensorSystemStateExpression :controlSensorSettingExpression
stepExecutionRules	:makeNextStepReady :conditionalStepExecution :labelStepExecution :gotoLabelStepExecution :funcallStepExecution :lambdaStepExecution :delayStepExecution :waitForStepExecution :assertIntoStepExecution :expressGoalStepExecution :returnFromGoalStepExecution :finishGoalStepExecution
systemGoalExpressionRules	:startupProcedure.goalExpression :controlInitiated.goalExpression :findObjectsOnTransferZone.goalExpression :returnInstrument.goalExpression
userGoalExpressionRules	:startProcedureGoalExpression :instrumentDeliveryGoalExpression :cancelGoalExpression
visionGoals	:calibrateBackground :findInstruments :findInstruments.preemptionHandler

Sensor expression refers to the consolidation of raw sensory input into a higher-level form more meaningful to the CA. For example one type of sensor expression might take image data from the cameras and "express" it in terms of the location of the viewed object, it's orientation, and any identification information that may have matched from the vision training data. It's a long process from the camera's raw pixel data to this high level characterization. Sensor expression rules are responsible for formatting only the essential information from this process for use in subsequent rules.

Sensor expression rules filter not only the amount of information as we've discussed, but also the frequency. Many of our sensors can provide their input at a high rate. The cameras for example

run at 15 frames a second. We don't want to bombard the CA with sensory input, so we use the sensor expression rules to provide the input at meaningful times. The vision sensor expression rules for example wait for motion to cease before analyzing the scene and presenting the new vision sensor input.

The next phase is goal expression. These rules decide if any action should be taken and, if so, they express a goal to accomplish that action. A goal expresses what is to be accomplished, like that an instrument should be delivered. Goals are divided into user-initiated goals and system initiated goals. The canonical example of a user-initiated goal is a goal to deliver an instrument to the surgeon. This goal is a direct result of user input. The analogous system initiated goal is a goal to return an instrument that has been lying on the transfer zone for a significant amount of time. This goal isn't based on any direct user input, but rather a system-initiated desire to keep the transfer zone clean.

Goal expression is usually triggered by sensory input, such as a voice recognition input naming a desired surgical instrument. But there may be several possible triggers for a goal's expression. For example, a prediction of the next needed instrument might trigger an instrument delivery goal instead of voice recognition. This is why we distinguish between sensor expression and goal expression. It provides a well-defined interface between the two concepts and allows the two modules to act independently.

Goal actualization is the next phase. Goal actualization rules are triggered by goal expression. They construct a list of things to do to accomplish the goal. If the goal is to deliver an instrument, this list would include steps to move the arm over the instrument, to turn on the magnet, to then move the arm to the drop-off location, and so on. We have constructed a rich language of step types to support sophisticated goal actualizations. These are essentially programs that the goal actualization rules write. Executing these programs will accomplish the goal. Goal actualization rules also often include a corresponding goal preemption actualization. This is a "program" to be run should the goal be cancelled or otherwise preempted in mid-execution. The instrument delivery goal preemption actualization, for example, will check to the see if the arm is currently gripping an instrument. If so, it will generate the steps required to lay that instrument down on the transfer zone as part of the cancellation clean up.

Goal actualizations are a list of "steps". The step execution rules are responsible for implementing each type of step. Some step execution rules initiate actuator output, such as moving the arm, speaking through speech synthesis, and even turning on LEDs. Other rules support conditional step execution, jumps within the step program, delays, and other useful programmatic tools.

Our validation process for the CA has included unit testing and integration testing. We have tested each rule in isolation by defining the system input requirements for the rule's trigger and the system state changes caused by the rule's action. We can then adjust the system inputs to test for proper triggering and introspect the system state knowledge base to verify the action. To accomplish this, we have developed extensive graphical user interfaces that allow us adjust and monitor any aspect of the system state knowledge base.

We have also been able to perform higher-level testing on the rule sets described above. As discussed, the actions of one set act as triggers for the next. Using the same graphical user interfaces, we are able to construct test cases that simulate the output of the actions of rules without actually using those rules. We can thus isolate the downstream rules for testing. For example, we can bypass the speech recognition sensor input rules by manually asserting that some utterance was detected. The speech sensor expression rules are not fired. In fact the hardware input device can even be disabled. This isolates this user goal expression rule from its usual input for replicable, careful testing.

Integrated testing is accomplished by using the actual robot – with all rules enabled – according to predefined test plans with expected results. As we provide input to the robot, we monitor its behavior and evaluate its performance. In addition we generate detailed rule trace output during validation testing which can be analyzed later to verify that expected rules were fired with the expected results. The rule trace is a timestamped list every rule firing with a human readable description of the rule's action. An example rule trace is shown below.

Instrument delivery rule trace output:

```
740: Actualize a goal to deliver the Richardson Retractor.
  77: Step: Speak phrase "Richardson Retractor".
  56: Step: Move arm from (0.275 0.0 -0.07) to (0.0 0.25 -0.075).
 137: Step: Finish waiting for properties (:motionTargetReached true) to appear.
 177: Step: Move arm from (0.275 0.0 -0.07) to (0.0 0.25 -0.115).
 173: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  98: Step: Set the magnet power to 0.8 (object on magnet: Richardson Retractor).
  99: Step: Call function (assertInto intrumentCacheState :cache :gripper).
  35: Step: Move arm from (0.275 0.0 -0.07) to (0.0 0.25 -0.055).
 187: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  32: Step: Set the magnet power to 0.8 (object on magnet: Richardson Retractor).
  31: Step: Move arm from (0.275 0.0 -0.07) to (0.5 -0.25 -0.04).
 144: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  80: Step: Call function (setState :waitingForBumpSwitch true).
 127: Running expiration timer for :waitFor to expire in 4 seconds.
2538: Cache new arm position, (0.5 -0.25 -0.04), in system state.
1554: Retracting expired :waitFor.
  12: Step: Skip :gotoLabel step due to failed execution condition.
  28: Step: Express goal :instrument.discard.
  29: Sorting goals according to priority.
  57: Actualize a goal to discard an instrument
  58: Step: Skip :gotoLabel step due to failed execution condition.
  48: Step: Move arm from (0.5 -0.25 -0.04) to (0.4288 0.1924 -0.12).
  84: Step: Finish waiting for properties (:motionTargetReached true) to appear.
 112: Step: Move arm from (0.5 -0.25 -0.04) to (0.4288 0.1924 -0.17).
  85: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  78: Step: Set the magnet power to 0.0 (object on magnet: nothing).
 114: Step: Call function (assertInto intrumentCacheState :cache :transferZone).
  25: Step: Move arm from (0.5 -0.25 -0.04) to (0.4288 0.1924 -0.08).
  47: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  21: Step: Ignore label :magnetIsEmpty.
 119: Step: Move arm from (0.5 -0.25 -0.04) to (0.275 0.0 -0.07).
 105: Step: Finish waiting for properties (:motionTargetReached true) to appear.
  11: Step: Acknowledge that goal :instrument.discard has been completely fulfilled.
  10: Step: Goto label :G547041312180974.
  94: Step: Ignore label :G547041312180974.
  42: Step: Call function (assert goals :goal :findInstruments :priority :system).
  18: Sorting goals according to priority.
 127: Step: Acknowledge that goal :instrument.delivery has been completely fulfilled.
```

We have thus been able to validate the behavior of the robot through a combination of isolated component testing and integrated system testing. The integration tests assure us that the robot can perform as required through a sequence of predefined use cases while the unit testing assures us the system reacts to any variety of inputs in a logical and predictable manner.

KEY RESEARCH ACCOMPLISHMENTS

- Completion of Software Development Tasks for the Penelope System

  Dual Camera Integration:
  Arm Masking Out:
  Auto Adjust Camera:
  Motion Detector:
  Multiple Simultaneous Object Identification:

- Completion of Hardware Tasks for the Penelope System\
  Design/Construction of Instrument tray and Transfer zone
  Design/Construction of System Stand:
  Hardware Integration

- Initial Validation of Entire System

REPORTABLE OUTCOMES

Manuscripts:
"Initial Clinical Experience With A Partly Autonomous Robotic Surgical Instrument Server"
Authors: Treat MR, Amory SE, Downey PE, Taliaferro DA
Publication: *Surgical Endoscopy.* This is a leading surgical journal, officially representing the Society of American Gastrointestinal Endoscopic Surgeons (SAGES) and the European Association for Endoscopic Surgery (EAES). It has a world-wide readership. The article will probably appear in print in the first quarter of 2006.

This article describes an actual surgery in which the Penelope Surgical Instrument Server assisted a surgical team in the performance of a simple surgical procedure. **The overall ability of the robot to perform successfully in this case was the direct result of the software and hardware tasks done in the work described in this Final Report.** The abstract from this article follows:

"Background: *We believe that it would be useful to have surgical robots capable of some degree of autonomous action in cooperation with the human members of the surgical team. We believe that a starting point for such development would be a system for delivering and retrieving instruments during a surgical procedure.*
Methods: *The robot delivers instruments to the surgeon and retrieves the instruments when they are no longer being used. Voice recognition software takes in requests from the surgeon. A mechanical arm with a gripper is used to handle the instruments. Machine-vision cameras locate the instruments after the surgeon puts them down. Artificial intelligence software makes decisions about the best way to respond to the surgeon's requests.* ***Results:*** *The robot was successfully used in surgery for the first time on June*

16, 2005. The operation was excision of a benign lipoma. The case lasted 31 minutes, during which time the robot performed 16 instrument deliveries and 13 instrument returns with no significant errors. Average time between verbal request and delivery of an instrument was 12.4 seconds.
Conclusions: *The robot is capable of delivering instruments to a surgeon on command and independently retrieving them using machine-vision. This robot, termed a "surgical instrument server", represents a new class of information processing machines that will relieve the operating room team of repetitive tasks and allow the team to focus more attention on the patient."*

Machine Vision For Surgical Robotics
Author: Treat, MR
Publication: *SPIE Biomedical Optics E-newsletter.* SPIE is a well respected international society for optical engineering.
URL: http://www.spie.org/newsletters/biomed/machinevision.html

This article is a discussion of the use of machine-vision to recognize and locate surgical instruments.

CONCLUSIONS

The overall purpose of this work is to develop a robotic device ("Robotic Scrub Technician or "Surgical Instrument Server") could perform some/all of the functions of the traditional scrub tech in a clinically acceptable way. This work began four years ago and with support from TATRC and others. During that time, we have steadily improved the capabilities of the machine. The work described in this report is a combination of software and hardware development and testing that contributes to the finalizing of the robotic scrub technician. The overall result of this work is that we have taken the final pre-clinical development steps toward achieving a clinical-grade robot, one that can perform in the actual operating room.

The **"so what"** of the work is several-fold. Such a device would be logistically better for forward deployed military hospitals. The device would be clinically better by reducing errors in counting/tracking instruments and supplies used in surgery. The device would improve overall operating room throughput by allowing personnel to be used in different ways to increase productivity.

A recent clinical study (see reference by Dr. Warren S. Sandberg), the Operating Room of the Future (ORF) Implementation Project, provides a blueprint for using Penelope SIS to improve operating room (OR) throughput and increase net operating room revenues for hospitals. This study, conducted by Dr. Warren Sandberg of Harvard University's Massachusetts General Hospital, is considered a landmark in the rational analysis of a chronic problem, that of achieving efficient use of expensive OR resources. This problem is all the more acute for hospitals today that must handle increasing numbers of surgical cases as overall reimbursement rates decline.

The study was based on the observation that non-operative processes occurring before and after the actual surgery are done *serially* in the typical OR. This serial workflow causes bottlenecks in the most time-consuming tasks, negatively affecting the entire workflow.

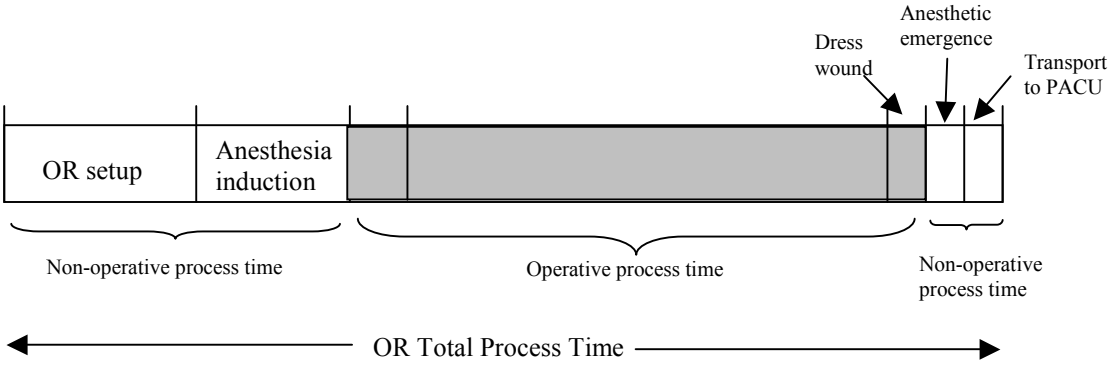

Fig. 9. Operating Room process time can be divided into non-operative and operative sections. A significant portion of Total Process Time is spent performing non-operative processes.

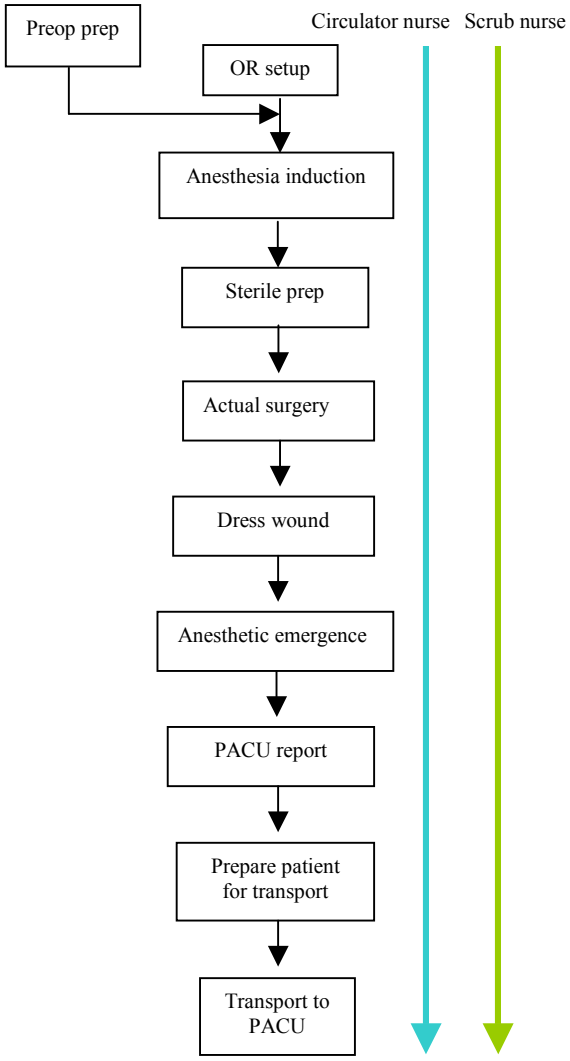

Fig. 10. In the standard OR, the workflow is essentially serial. All non-operative and operative processes are done one after another. Standard staffing is a circulating nurse and a scrub nurse or tech.

The ORF Implementation Project reorganized non-operative processes so they are performed in parallel with operative processes. The study's outcome demonstrated an increase in overall OR throughput due to a major reduction in non-operative time between cases. The study showed that, for common general surgical cases, the timesavings allowed for one extra surgical case without running into expensive overtime hours. A 2003 article in Healthcare Financial Management stated that "improving throughput by just one additional procedure per day per OR suite can generate anywhere from $4 million to $7 million in additional annual revenue for the average-sized organization."

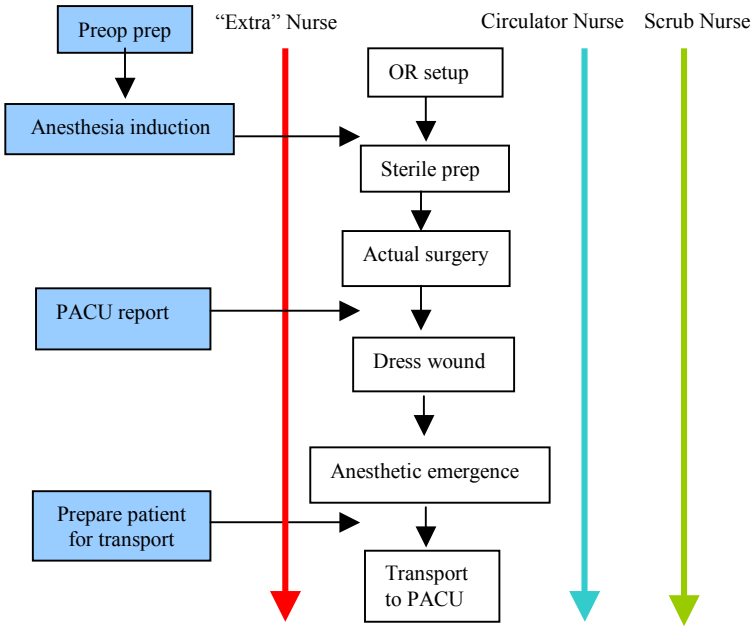

Fig. 11. In the reorganized OR, significant non-operative processes are done in parallel (shown in blue). This shortens the overall OR process time but does require extra staff (red arrow). OR throughput will be increased but this may be revenue neutral due to additional staffing costs.

However, the increased throughput entailed additional costs. These costs were mostly due to the need for an extra nurse to perform the non-operative processes in parallel. This resulted in a 13% increase cost in the ORF model (parallel process) over that of the typical OR model (serial process).

According to the primary architect of this study, Dr. Warren S. Sandberg, the Penelope robot would be the ideal tool to allow the OR to achieve increased throughput through parallel processing, while maintaining personnel costs at the level found in the slower traditional OR's. The robot would allow one Operating Room nurse to perform the functions of both circulator and scrub nurses, since the robot would offload the time-consuming and labor intensive parts of the scrub nurse's job. That would free up a Full Time Employee (FTE) nurse to perform the parallel processing and reduce non-operative time between cases.

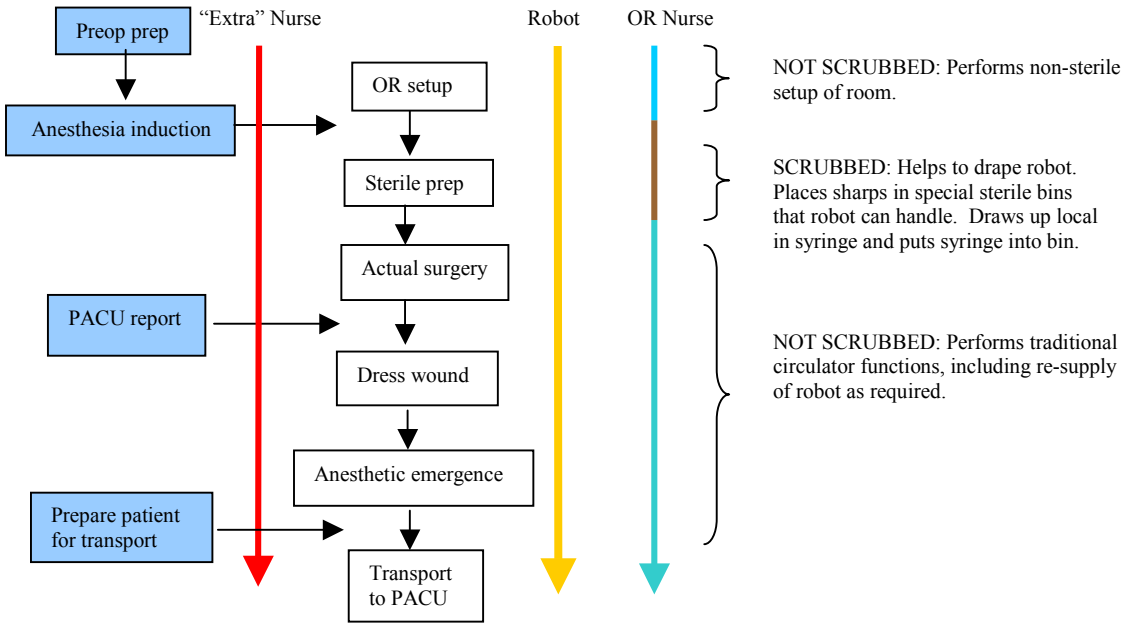

Fig.12. The robot enables parallel performance of non-operative tasks while keeping nursing personnel requirements at two FTE's. This produces the increased workflow of the parallel processing OR without the additional personnel costs.

The Penelope robot allows one OR nurse to perform the circulator functions as well as the few scrub functions that the robot does not do, such as set up medications and set up sharps. This keeps the overall FTE requirement to two (like the traditional OR serial processing) while maintaining the advantages of parallel performance of significant non-operative processes.

REFERENCES

Machine Vision For Surgical Robotics, Treat, MR
SPIE Biomedical Optics E-newsletter
URL: http://www.spie.org/newsletters/biomed/machinevision.html

Deliberate Perioperative Systems Design Improves Operating Room Throughput. Sandberg WS, MD, PhD, Daily B, MHA, Egan M, RN, MS, Stahl JE, M.D., CM, MPH, Goldman JM, MD,_Wiklund RA, MD, Rattner D, MD. Anesthesiology 2005; 103:406–18

"Achieving Operating Room Efficiency Through Process Integration". Healthcare Financial Management, March 2003

APPENDIX A: LIST OF PERSONNEL

Fig. 13. Left to right: Michael Treat, Michael Brady, Stephen Leonard, Jay Klein, Patrice Downey, Russell Baker

Michael R. Treat MD, President and CEO

Dr. Treat is working full time at Robotic Surgical Tech, Inc. He provides overall leadership for the group. His support for the group includes technical assistance, clinical input, securing financial resources, and interfacing with the rest of the world. He was the original designer of the first Penelope system prototypes and is intensely involved in all aspects of the robot development. He is titled as Associate Professor of Clinical Surgery, College of Physicians & Surgeons of Columbia University and Attending Surgeon at the NewYork-Presbyterian Hospital. Dr. Treat has numerous patents in the area of surgical devices. Two of his surgical device inventions achieved commercialization. The first was a bipolar electrocautery snare for endoscopic polypectomy, which employed a new approach to improving the safety of endoscopic polyp removal. This device achieved only limited commercial success but another company subsequently imitated the concept more successfully. A startup company in San Jose California, Starion Instruments Corporation is currently marketing another of his devices, a thermal tissue-welding device. Starion Instruments is a success story with growing sales using the thermal tissue welding technology invented by Dr. Treat and licensed to Starion by Columbia University. Dr. Treat has lectured three times on surgical device development at the Executive MBA class taught by Adjunct Business School Professor Jack Kaplan at Columbia University Business School.

David Michael Brady, Chief Technical Officer

Mr. Brady is the chief technical architect of the system. Besides software design, he is also the main architect of the overall system including its physical layout and overall mechanical design. He joined the project at an early stage of development. He was responsible for engineering the extensive Java language code base of the robot, achieving major performance increases that allowing the application to run comfortably on a standard laptop computer. The current robustness of the Penelope code is due to his excellent workmanship. Among Mike's professional experiences was his job as Lead Designer and Developer, Boeing Missiles and Space Division, Advanced Computing Group. There he lead the development team for CPACS, a cargo planning and analysis tool for the Space Station. CPACS is used to configure cargo for all space shuttle resupply flights to and from the Station. It was delivered ahead of schedule and under budget. Not only was he the primary software designer and developer but he developed strong leadership skills. As a result of this work, Mike won the Manned Flight Awareness Team Award, which is NASA's highest civilian award for meritorious work contributing to the space program.

Stephen Leonard, Robot Engineer

Mr. Leonard is in charge of all aspects of the counting system that Penelope uses to keep track of surgical instruments and other items used in surgery. He is in charge of integrating the machine-vision system and the AI capability of the robot into a robust counting system that is capable of recovering from errors. Mr. Leonard graduated with a BS and MS in Computer Science from New York University. While at NYU, he worked in for data acquisition and experimental control of neurophysiological experiments including the running of Cortex for laboratory research includes designing and implementing experiment specific training and testing programs (in a C subset language) as well as managing various input devices e.g. Touch Screen, Eye Scan. He also worked as a software engineer at Thinkmap, Inc. developing enterprise-level software applications for the management and sharing of complex digital information.

Jay Klein, Robot Engineer

Mr. Klein's primary area of responsibility is the computer vision system, and he also assists Mr. Brady in general AI tasks. He also participates in all other aspects of Penelope's design. Mr. Klein attended the University of Michigan, taking part in both the Residential College and the Honors Program, concentrating in Artificial Intelligence within Computer Science. He graduated with a BS in Computer Science in 2003. While there, working with Professors Elliott Soloway and Layman Allen, he developed a version of the math game EQUATIONS to run on Pocket PCs using Bluetooth technology for wireless multiplayer capability. He also studied extensively in the field of Natural Language Processing, developing a natural language search engine for the Internet Movie Database and modifying the language realizer FUF/SURGE to produce text with meter and rhyme for poetry generation. Since joining Robotic Surgical Tech, Inc. in June of 2003, he has assisted in the design of a cognitive AI architecture for the robot. He developed a test version of a sequence based Prediction Engine which helped in the obtainment of an NSF Small Business Innovation Research grant for the company.

Patrice E. Downey RN BSN, Vice President & COO

Ms. Downey has over 20 years of business experience covering all phases of surgical device development and marketing. Before joining RST in June of 2004, she was Vice President of JARIT Surgical Instruments, Inc., a leading surgical instrument manufacturer. Ms. Downey was

with JARIT for over 12 years before she decided to strike out on her own and join a start-up company, Robotic Surgical Tech, Inc. Prior to that, she held positions in marketing, product development, training and sales for Cabot Medical, Inc. and Support Systems International, a Hillenbrand Industries, Inc. company. In addition, Ms Downey spent 8 years in the nursing profession in clinical practice and management positions. Ms. Downey is a cum laude graduate of Niagara University, Niagara Falls, New York.

Russell Baker, Robot Engineer

Mr. Baker's primary area of responsibility is in the design and building of all electromechanical systems. His skills and responsibilities include prototype making using CAD, machine tools, composite-construction techniques such as epoxy resin-fiberglass-foam, welding and adhesives. He is also responsible for assembly language programming of the hardware controllers and sensors. He also participates in all other aspects of Penelope's design. Mr. Baker graduated from Columbia University School of Engineering and Applied Science in the spring of 2002 with a B.S. in Mechanical engineering. He studied Robotics, Geometric Modeling, and Data Structures for Java. During his final semester, he designed and built an eight-foot robotic arm for Prof. Peter Allen of the Computer Science department. The arm was intended to hold an underground scanning radar that could not be near any metal; hence the robot was entirely nonmetallic. His practical experience in working with non-metallic materials to build a robot arm has been ideal for the Penelope project, since one of the design requirements is that the robot arm be lightweight and in fact is made largely of carbon-fiber.

APPENDIX B: REVIEW OF PENELOPE PROJECT

The work described in the Final Report for this project (W81XWH-05-1-0410 "Pre-Clinical Validation of Robotic Scrub Technician") should be taken in the context of the overall "Penelope Project". We have received support for this project from TATRC since 2003 (DAMD17-03-C0083 "Robotic Replacement for Surgical Scrub Technician"). While the work described in the Final Report is for specific technical software and hardware tasks, it is fitting to review the fruits of this work in terms of public recognition and national scientific recognition that have accrued to the Penelope Project as a whole.

Overview of the Penelope Surgical Instrument Server

The Penelope robot is designed to perform some of the functions of the scrub technician in the surgical operating room, as shown in the following figure.

Fig. 14. Basic actions of the Penelope robot. The surgeon requests an instrument and the robot hands off that instrument. The surgeon lays the instrument down when he/she is finished with it, and the robot retrieves it (3) and returns it to the instrument tray (4).

The overall hypotheses of this project are:

- A robotic device ("Robotic Scrub Technician or "Surgical Instrument Server") could perform some/all of the functions of the traditional scrub tech in a clinically acceptable way.
- Such a device would be logistically better for forward deployed military hospitals.
- Such a device would be clinically better by reducing errors in counting/tracking instruments and supplies used in surgery.

- Such a device would improve overall OR throughput by allowing personnel to be used in different ways.

Penelope is the world's first autonomous, machine-vision guided, robotic surgical assistant. "She" is a co-worker or helper for the operating room staff. Penelope uses voice recognition to respond to the surgeon's verbal request for any of the variety of instruments required during a surgical procedure. Drawing from an assortment of instruments that she manages on an instrument tray, Penelope hands the instrument to the surgeon with a robotic arm. Using a visual capability, the robot then locates any instrument that the surgeon has finished using and returns it to the instrument tray. She can also count and document instruments and other items used in the surgical operation.

One of the things that makes this robot so different from other surgical robots is that the overall behavior of the Penelope system is controlled by its software "cognitive architecture". This is artificial intelligence software that enables the robot to respond in a clinically appropriate manner to the changing environment within the OR. The cognitive architecture also gives the robot the ability to recognize errors both human and robotic, and then respond with corrective action. The cognitive architecture contains a rule-based inference engine and a set of rules that determine the robot's behavior. These rules are essentially IF-THEN statements. Inputs to the cognitive architecture come from the robot's sensory subsystems such as the visual and auditory subsystems. Inputs also come from sensors which monitor the internal state of the robot, such as position sensors in the joints of the arm. In the cognitive architecture, the inference engine constantly scans all of the inputs and compares them to the IF clause of all the rules. When the conditions of an IF clause of a rule are met, the rule "fires", producing some sort of output of the robot. Outputs may be motor actions, speech utterances or internal "assertions" that are presented back to the inference engine for further consideration. With a fairly large number of rules, very complex behavior can be produced, potentially giving the robot great flexibility to respond to varying conditions including off-nominal ones. Off-nominal conditions could potentially be caused by human or robotic errors. The cognitive architecture will contain rules that set goals (a "goal stack") that must be met when the robot is performing some task. Essentially, these "goal stack" rules will require the robot to check on the integrity of the task process, and to branch into other actions if the a goal is not met. For example, a request to delivery an instrument will generate assertions commanding the motor system to do the appropriate things (such as pick up the instrument), but will also cause the firing of a rule that will set up a goal stack for that delivery action. If the sensors (i.e. the vision system) report to the cognitive architecture that the instrument is unexpectedly no longer in the gripper, this will trigger the firing of rule which essentially states "If the instrument is no longer in the gripper when it should be in the gripper, then go and look for it." This type of goal seeking means that the robot will be quite relentless in trying to carry out intended actions.

Another feature of the software is the "prediction engine". This is a software construct that uses statistical techniques to anticipate the surgeon's requests for instruments. The software creates a database about a surgeon's individual preferences. The system has the ability to build upon its ongoing experience to continually learn and improve. The outputs of the prediction engine are used by the cognitive architecture as another type of input that helps determine the robot's actions.

The Penelope system will enable hospitals to increase the effectiveness of nursing and technical personnel in the operating room, and possibly redistribute staff coverage to currently underserved areas in the hospital. The following statements are from the leadership of the NewYork-Presbyterian Hospital and the Department of Surgery:

"Health care faces diverse issues of quality, access and cost, many of which can be addressed by appropriate application of technology. This project is part of our continued commitment to the advancement of medical technology for the better treatment of the patient and the greater efficiency of the hospital."
Dr. Herbert Pardes, President and CEO of NewYork-Presbyterian Hospital.

"As in other areas of health care, the surgical team has faced shortages of manpower that affect the scheduling and staffing of operations. The Penelope SIS has the potential to help extend the operating room staff, allowing more versatility in the scheduling of routine procedures and automation of instrument counts. It can be an innovative tool in our work to improve the quality of patient care."
Wilhelmina Manzano, M.A., R.N., Senior Vice President and Chief Nursing Officer at NewYork-Presbyterian Hospital

Evolution of Penelope

Penelope 0, September 2001

Penelope 1.0, April 2002

Penelope 2.5, April 2003

Penelope 2.8, June 2004

Fig. 15. Serial Penelopies

In developing the robot, **Robotic Surgical Tech, Inc.** has received grants from the **National Science Foundation** ($600,000) and the **National Institutes of Health** ($100,000), and research & development contracts from the **Telemedicine and Advanced Technology Research Center** of the US Army ($133,000) and the **Defense Advanced Research Projects Agency** ($250,000).

Penelope 0, September 2001, is shown in Fig. 14 above, top left. This was a cardboard mock-up which was built in order to test out the basic shape and size of the planned machine. It was actually taken to the Allen Pavilion OR where it was set on a Mayo stand and its reach and range of motion were checked. The basic dimensions and shape have essentially been retained right up through the present clinical version of the robot.

Penelope 1.0, April 2002, is shown, top right. This was the first working version of the robot. It was constructed by Dr. Treat and Martin T. Lichtman, a young mechanical engineer who had just graduated from Brown University. The robot was physically built by the two of them, and all of the computer code was written by them. The robot had a primitive digital camera connected to the main computer (a PC laptop) which performed identification and localization of three surgical instruments. The arm, constructed of aluminum tubing in an innovative truss design,

was actuated by hobby servos controlled by a microcontroller that communicated with the main computer. The arm used an electromagnet to pick up the instruments, a design feature which remains to this day. This robot, though somewhat shaky and awkward was good enough to get our first round of external funding, from the US Army's Telemedicine and Advanced Technology Research Center (TATRC) who saw the possibility of it being used in forward deployed military hospitals.

Penelope 2.5, April 2003, is shown bottom left. This machine was a major software upgrade and total physical rebuilding of the design used in the original working prototype. It had all metal construction of the arm and many enhancements to the machine vision and motion control software. It was mounted in a fiberglass base which was handmade by the team, and shown at the TATRC exhibit at the 2003 American Telemedicine Association meeting. It drew a lot of interest from visitors.

Penelope 2.8, June 2004, is shown bottom right. This machine was meant to embody all of the features that would be necessary to perform in the surgical operating room. The basic concepts of having a mechanical arm and an overhead machine-vision camera were retained from previous versions, but the overall physical layout was completely new. The layout was carefully designed to handle the number of instruments needed to support a general surgical case, and included special instrument trays which could be detached for sterilization. It also featured a "vertical back tray", which was intended to hold additional instruments. This machine performed in a mock-surgery in an operating room at the Milstein Hospital, as part of the John Jones Surgical Society Day, in June 2004.

Much of the early work was done in space provided by the NewYork-Presbyterian Hospital, at the Allen Pavilion. Later, as the project grew, the work was relocated to a space which was a former Women-Infants-Children (WIC) Clinic run by the Hospital.

This work culminated in Penelope 3.0 Alpha, the first clinical grade robot of the series. This robot has the basic capabilities to work in the operating room. It will serve as a clinical test bed and technology platform for the next robot in the series, Penelope 3.0 Beta, the "commercial grade robot" which will be mass produced. This machine was designed and built entirely by the team of Robotic Surgical Tech, Inc. The machine was completed in March, 2005 and performed in her first clinical case shortly thereafter.

Fig. 16. Penelope 3.0, outfitted for surgery and wearing logos of supporters.

First Clinical Case

Fig. 17. The Penelope robot scrubbed in for "her" first clinical case on June 16,2005 at the Allen Pavilion of the NewYork-Presbyterian Hospital. Dr. Spencer E. Amory, Director of Surgery at the Allen Pavilion, performed the case, which was the excision of a benign tumor (lipoma) from the patient's forearm. Penelope performed her assistant role successfully.

The following was typical of the many news reports that followed:

NY1 NEWS
NY-Presbyterian Uses Robot To Assist In Surgery For The First Time June 16, 2005

Doctors at New York-Presbyterian say you haven't seen anything like this. As NY1's Kafi Drexel reports, for the first time ever, it's not just nurses helping out in the operating room, now they're relying on some hi-tech assistance.

Meet Penelope. She's now giving doctors at New York-Presbyterian an extra hand, at least that's how the doctors might put it.

Penelope is really a robotic arm, quipped with voice recognition technology that understands commands from doctors and nurses in the operating room. A surgeon can ask for any instrument, a scalpel for example, and hand it over.

Not only that, Penelope also has software that can predict what kind of instrument a surgeon might need next. She even talks!

“It's the 21st century, and we should have machines like this helping out in surgery,” says Dr. Michael Treat, the creator of Penelope. “There's plenty of work in surgery. You may not realize how many things we have to keep track of.”

The robot, known as the Penelope Surgical Instrument Server, was designed by Dr. Treat, founder of Robotic Surgical Tech of New York. In a medical first Thursday, the robot assisted in the removal of a benign tumor on the forearm of a patient.

Doctors and nurses say not only does the robot help save time in the OR, but they say Penelope is a valuable tool that comes at a time when surgical teams are facing shortages in manpower.

Penelope does a lot in the operating room, but most likely she won't be putting doctors and nurses out of their jobs any time soon. They say the robots aren't there to replace them, but to simply make their jobs easier.

“We envision that Penelope will relieve us, the surgical team, of some of the repetitive tasks that would otherwise consume a lot of our time and energy,” says Dr. Spence Amory of New York-Presbyterian. “This will free us up to remain focused on patients and the surgical procedure.”

While the robot's only there to assist the doctors, and never has direct contact with patients, doctor's say Penelope and other technology like her is something patients will start seeing a lot more of operating rooms.

They do admit that the robot has made mistakes, but she's always supervised by a human to correct her. Perhaps that's why they believe even though the robot helps, she'll never replace the human touch.

- Kafi Drexel

National Museum of Health & Medicine

Fig. 18. New exhibit dedicated to the Penelope Surgical Instrument Server at the National Museum of Health & Medicine in Washington DC

The **National Museum of Health and Medicine** is a branch of the **Armed Forces Institute of Pathology** and is located in Washington DC on the campus of the Walter Reed Army Medical Center. The National Museum of Health and Medicine was founded as the Army Medical Museum in 1862 to study and improve medical conditions during the American Civil War. The Museum houses a collection of over 24 million items including archival materials, anatomical and pathological specimens, medical instruments and artifacts, and microscope slide-based medical research collections. The collections focus particularly on the history and practice of American medicine, military medicine, and current medical research issues.

The following announcement was recently put out by the Museum.

"**National Museum of Health and Medicine announces opening of new robotic Surgical Instrument Server exhibit**

The National Museum of Health and Medicine has announced the opening of its newest exhibition -- "Penelope: The World's First Autonomous, Vision-guided, Intelligent, Robotic Surgical Instrument Server."

A robotic scrub assistant with speech recognition, machine vision, and robotic arm path planning and targeting, Penelope was developed by Robotic Surgical Tech, Inc., a Columbia University spin-out enterprise.

Michael R. Treat, M.D., president of the Penelope Team, demonstrated the first functional prototype in 2002 at the "Engineering the Future of Surgery" symposium sponsored by the Telemedicine and Advanced Technology Research Center (TATRC) of the U.S. States Army. Robotic Surgical Tech, Inc. receives support from the Army and TATRC.

Penelope is comprised of 4 major hardware and software components: the robotic arm, the instrument platform, the system stand, and the system control software.

"Penelope has the potential to save thousands of dollars each year and free up valuable hospital operating room staff for other tasks," Treat said. "We're thrilled that our prototype will be immortalized through its acceptance into the collection at the National Museum of Health and Medicine."

http://nmhm.washingtondc.museum/news/robotic_surgical.html

World Technology Evaluation Center Citation

The World Technology Evaluation Center recently cited the Penelope robot as an example of US leadership in medical robotics. On September 16, 2005, the findings of an *International Study of Robotics by the World Technology Evaluation Center* were made public. The report culminates a nearly 2-year effort to evaluate robotics research and development in the United States, Japan, Korea and Western Europe. The study was sponsored by the National Science Foundation, NASA and the National Institutes of Health. The study report was produced by a panel of six robotics experts who visited more than 50 research sites across the globe. The detailed report findings were presented at daylong workshop presented. Panel chair George Bekey, Professor of Computer Science from the University of Southern California, presented a summary of the results. A recorded webcast version of the summary presentation is available at http://www.nsf.gov/news/newsmedia/robotics05/index.jsp

As Dr. Bekey mentioned, the findings for the United States are not all positive. U.S. researchers have developed advanced robotics, but national strategies and coordinated funding efforts in other countries pose a serious challenge. Strong competition from Japan, Korea and Europe has overtaken the United States in many areas.

The report did state however, that the United States leads in the area of robot-assisted surgery. At the presentation, Dr. Bekey showed a slide containing a photo from Penelope's first surgery at NewYork-Presbyterian Hospital. Referring to the slide, Dr. Bekey said, "On the right hand side [of the slide] is a fairly new robot, which is known as—who is known as Penelope. Penelope is a surgical assistant that can hand tools to a surgeon." The slide contained two other photos: a telepresence robot that transmits pictures of a patient to a remotely located physician and the da Vinci surgical robot. Dr. Bekey stated, "We're the most active in robotic surgery and the development of tools and assistance for medicine."

The Penelope Surgical Instrument Server was developed by Robotic Surgical Tech, Inc., at the NewYork-Presbyterian Hospital with the support of the NewYork-Presbyterian Hospital, the National Science Foundation, the Telemedicine and Advanced Technical Research Center of the United States Army, the Defense Advanced Research Projects Agency and the National Institutes of Health.